Expert Systems with Applications 40 (2013) 231–241

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Algorithm learning based neural network integrating feature selection and classification

Hyunsoo Yoon [a], Cheong-Sool Park [b], Jun Seok Kim [b], Jun-Geol Baek [a,*]

[a] School of Industrial Management Engineering, Korea University, Seoul, Republic of Korea
[b] Graduate School of Information Management and Security, Korea University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Feature selection
Classification
Neural network
Extreme learning machine
Algorithm learning based neural network (ALBNN)

ABSTRACT

Feature selection and classification techniques have been studied independently without considering the interaction between both procedures, which leads to a degraded performance. In this paper, we present a new neural network approach, which is called an algorithm learning based neural network (ALBNN), to improve classification accuracy by integrating feature selection and classification procedures. In general, a knowledge-based artificial neural network operates on prior knowledge from domain experience, which provides it with better starting points for the target function and leads to better classification accuracy. However, prior knowledge is usually difficult to identify. Instead of using unknown background resources, the proposed method utilizes prior knowledge that is mathematically calculated from the properties of other learning algorithms such as PCA, LARS, C4.5, and SVM. We employ the extreme learning machine in this study to help obtain better initial points faster and avoid irrelevant time-consuming work, such as determining architecture and manual tuning. ALBNN correctly approximates a target hypothesis by both considering the interaction between two procedures and minimizing individual procedure errors. The approach produces new relevant features and improves the classification accuracy. Experimental results exhibit improved performance in various classification problems. ALBNN can be applied to various fields requiring high classification accuracy.

1. Introduction

During the last decade in the bioinformatics, the advent of microarray techniques has led to the utilization of sufficient information regarding the condition and illness of patients (Sarkar et al., 2002). The right diagnosis is crucial for the proper treatment of patients. Thus, statistical methods for determining the salient features affecting the condition of patients are being explored. Moreover, in chemical engineering or semiconductor manufacturing processes, the quality of a product is determined by the interaction between multiple attributes of the production process or characteristics of each attributes. The use of statistical methods for determining several important features affecting the quality of a process has also helped to reduce the production of defective products at an early stage by blocking them and increase cost-effectiveness.

The classification of multivariate data, the foundation for such statistical analysis, has been explored in two phases: feature selection and classification. Feature selection reduces the dimensions of or extracts relevant features from an original dataset. The classification procedure classifies raw data or selected features using classification algorithms. However, feature selection and classification techniques have been studied independently. The interaction between both procedures has not been thoroughly explored. In this study, the main objective is to improve classification performance. Therefore, we propose a new method that can integrate the two procedures using neural networks and statistical learning approaches to minimize classification errors.

Feature selection techniques have two kinds of approaches depending on the use of the target information: unsupervised learning and supervised learning methods. In unsupervised learning, a principal component analysis (PCA) or an independent component analysis (ICA) considers correlations between variables or properties and alters the original representation to extract features without a significant loss of information. However, no information exists about the target values, which may not be suitable for implementation in classification (Park & Choi, 2009). On the other hand, supervised learning approaches employ class information, which is known beforehand. Filter techniques assess the relevance of features by considering the intrinsic characteristics of the data, whether the current feature is relatively important or not. These methods are refined limitations of unsupervised learning and are computationally fast, but they present issues, such as ignoring feature dependencies and disregarding the interaction between selection and classification algorithms (Saeys, Inza, &

* Corresponding author. Tel.: +82 2 3290 3396; fax: +82 2 929 5888.
E-mail address: jungeol@korea.ac.kr (J.-G. Baek).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.07.018

Larranaga, 2007). The wrapper approach, a supervised learning method, is an augmented means to include the interaction between the feature subset and classification algorithm. This selection approach defines the space of the possible feature subset, and various subsets of features are sequentially generated in a forward or backward direction as their performance compares with the former. However, in this approach, the space of feature subsets grows exponentially with the number of features, heuristic search methods are required to search for an optimal subset, and the approach is very computationally intensive. Moreover, it exhibits over-fitting problems (Kabir, Islam, & Murase, 2010; Pudil, Novovicova, & Kittler, 1994). In the study of classification, a large number of statistical techniques have been developed to classify original data or selected features for suitable targets. Support vector machines (SVMs), developed by Cortes and Vapnik (1995), have become popular classification techniques. SVMs maximize the margin, which is either side of a hyper plane that separates two data classes, and thereby create the largest possible distance between the separating hyper planes. In multivariate classification, SVMs determine support vectors (SVs) in a reduced dimension using selected features. However, if the selected features to be used for classification do not reflect any important information, SVMs cannot determine the hyperplanes in the correct directions, causing a downgrade in the performance of the SVMs (Burges, 1998).

Artificial neural networks (ANNs) are also widely used to estimate complex target functions through suitable modelling and iterative learning. They attempt to determine a nonlinear function based on the network's architecture without the constraint of linearity and assumptions. Additionally, due to the improving performance of computer systems, these networks can be easily applied to utilize and estimate models in various areas with complexity. Therefore, neural networks have been used as an alternative to conventional statistical techniques, such as regression and discriminate analysis, in the recent decade (Paliwal & Kumar, 2009). Neural networks represent more powerful and adaptive nonlinear equation forms. They are able to learn complex functional relationships between the input and output data (Kimes, Nelson, Manry, & Fung, 1998). Due to these characteristics, if the output nodes contain real values, they have the capacity to be a regressor. Furthermore, a classifier can possibly be defined as when the outputs are integer or categorical values. In general, the feature selection techniques provide real values as outputs and the classification algorithms return categorical values. Therefore, we employed neural networks in order to estimate the traits of feature selection and classification.

Using neural networks presents some problems. When those are utilized, determining the structural design, the activation function, and learning methods are crucial issues for good performances (Egrioglu, Aladag, & Gunay, 2008). Even though various approaches have been proposed to determine the best structure of neural network in the literature, the trial and error approach is the most preferred yet. Cagdas (2011) proposed selection method based on tabu search algorithm to find best architecture. However, those approaches require time-consuming work. Another system structures also have been explored by the hybrid approaches (Sahin, Tolun, & Hassanpour, 2012). In this study, we consider a unique method to avoid the time-consuming process of determining the best model selection employing ELM algorithm and integrating two learned neural networks with consistent performance. The techniques regarding the model selection are described in more detail in Section 2.

In general, neural networks provide variable results, and the learning speeds depend on the establishment of random initial points. Fig. 1 illustrates a hypothesis search space and range by different initial points and learning methods. The learned neural networks are commonly distributed throughout the improperly wide

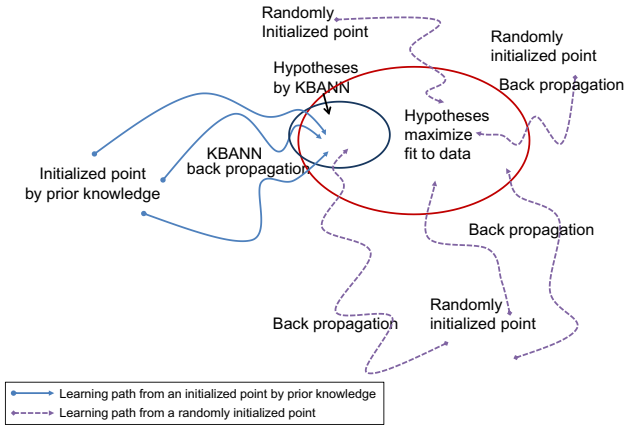

Fig. 1. Hypothesis space search in general NN and KBANN.

regions in the hypothesis spaces for estimating target function. To fit the target hypothesis more suitably, one approach is to initialize the network using domain information as the prior knowledge. This approach is described in (Towell & Shavlik, 1994) and they termed this approach to initialize a neural network as knowledge-based artificial neural networks (KBANNs). A KBANN operates on prior knowledge from domain experience, which provides it with a better starting approximation of the target function than the initialization of the network from random initial weights. This leads to better generalization accuracy and convergence into more specific hypothesis spaces, as shown in Fig. 1. This study is based on these effects of a KBANN, which improve the performance of classification.

Although a KBANN analytically creates a network equivalent to the given prior knowledge, a KBANN can possibly be misled when highly inaccurate domain knowledge is given. Moreover, prior knowledge is usually difficult to identify. Therefore, we utilize prior knowledge that is mathematically calculated from the properties of other learning algorithms instead of using unknown background resources.

The remainder of the paper is organized as follows. In section 2, we describe the proposed method with an extreme learning machine, discuss its effects, and illustrate a simulation study. In section 3, we compare this study with conventional methods using real world datasets. In section 4, we conclude this study with suggestions for future research.

2. Algorithm learning based neural network (ALBNN)

A learning algorithm is used to determine the best weights for an estimating target function in neural networks. The back-propagation algorithm has been widely used for learning in neural networks. These methods employ a gradient descent to update weights iteratively, minimizing errors between the network outputs and the target values (Chauvin & Rumelhart, 1995). Several back-propagation algorithms, such as the gradient-based algorithm, adaptive method, quick prop, and second-order methods, exist. However, these can become trapped in the local minima of the error surface, and the training performance varies according to the algorithm parameters (Wang & Huang, 2007). Among them, the Levenberg–Marquardt (LM) method performs well in solving nonlinear optimization problems (Hagan & Menhaj, 1994). Even though the LM algorithm has good performance in the optimization field, it has some limitations when used in a neural network. The performance of a neural network depends highly on the architecture determined manually by trial and error; however, the best

architecture is difficult to determine. We implement both an extreme learning machine (ELM) and LM algorithm in the learning of the proposed neural network approach to avoid the limitations.

2.1. Extreme learning machine (ELM)

An ELM is an emergent statistical learning method that has performed well in regression and classification. This technique was developed for single hidden-layer neural networks that exhibited some limitations with manual tuning and dependency on the architecture (Huang, Zhu, & Siewa, 2006). The core of ELM does not need to tune the weights with randomly assigned input to the hidden layer weights, in addition to analytically determining the output weights from the hidden layer. By estimating the hidden output weights using a simple mathematical approach, it avoids the issue of identifying the best parameters with iterative tuning. An ELM employs the Moore-Penrose generalized inverse technique of a weight matrix and then estimates by simple least-squares (Huang, Zhou, Ding, & Zhang, 2011). The simple ELM procedure is as follows:

Given a training set: $N = \{(x_j, t_j) | x_j \in R^d, t_j \in R^m, j = 1, \ldots, N\}$

Step 1. Randomly generate input to hidden weight vectors $w_i = (a_i, b_i)$ $i = 1, \ldots, l$ l: hidden node number N

Step 2. Calculate the hidden layer output matrix H

$$\sum_{i=1}^{l} \beta_i g_i(w_i x_j + b_i) = t_j, \quad j = 1, \ldots, N \tag{1}$$

$$\mathbf{H}(w_1, \ldots, w_l, b_1, \ldots, b_l x_1, \ldots, x_N) = \begin{pmatrix} w_1 \cdot x_1 + b_1 & \cdots & w_l \cdot x_1 + b_l \\ \vdots & \ddots & \vdots \\ w_1 \cdot x_N + b_1 & \cdots & w_l \cdot x_N + b_l \end{pmatrix} \tag{2}$$

Step 3. Calculate the output weight vector β

$$\mathbf{H}\beta = \mathbf{T} \tag{3}$$

$$\|\mathbf{H}\hat{\beta} - \mathbf{T}\| = \min_{\beta} \|\mathbf{H}\beta - \mathbf{T}\| \tag{4}$$

$$\bar{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{5}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of the matrix.

Compared to conventional learning techniques for feed forward neural networks, an ELM provides slightly improved accuracy performance and a significantly faster learning speed. Furthermore, the performance of an ELM is not sensitive to the number of hidden nodes, unlike the results of general back-propagation learning. Therefore, we implement an ELM in this study to determine the neural network architecture and calculate the unknown background knowledge.

2.2. ALBNN procedures

The theory behind a KBANN is the basis for the algorithm learning based neural network, which we have termed ALBNN. Although domain knowledge can pertinently initialize the network, prior knowledge is usually difficult to identify in many fields. However, instead of using unknown background resources, an ALBNN incorporates estimated prior knowledge obtained from the properties of other learning algorithms, such as feature selection and classification algorithms. Yoon and Baek (2011) previously designed this approach but observed some limitations in appropriately estimating prior knowledge. In the learning procedure using back propagation, they used an LM algorithm to calculate the traits of prior knowledge; however, the algorithm requires time-consuming work to determine the best architecture, and the learning time is tedious. If the algorithm determines the best weights, an over-fitting problem may occur or the local minima may become trapped (Paliwal & Kumar, 2009). Unlike back-propagation learning using an LM algorithm, an ELM requires much less learning time and has good generalization accuracy, thereby avoiding the local minima issue. Therefore, we employ an ELM to estimate the prior knowledge in order to avoid the previous limitations of an ALBNN. This procedure helps to obtain better initial points faster and bypass irrelevant time-consuming work, such as determining the architecture and manual tuning.

Following Fig. 2 shows the ALBNN learning steps. Three steps are conducted to build the proposed model. In the first and second steps, the approximated prior knowledge is learned using an ELM, and the neural network architecture is automatically determined as they process. The third step combines the two kinds of NNs and learns using an LM algorithm. Through these steps, the ALBNN model provides improved performance for the classification problem.

The ALBNN's procedure is as follows:

1st step: Create first neural network that learns the properties of the feature selector using an ELM.
2nd step: Create second neural network that mimics the functions of the classifier using an ELM.
3rd step: Incorporate the first and second neural networks and learn using an LM.

2.2.1. Create first neural network that learns the properties of the feature selector using an ELM

In this step of the ALBNN procedure, the single hidden-layer neural network learns the characteristics of the existing feature selection algorithm by itself. Initially, one conventional feature selection algorithm, such as the PCA or least angle regression (LARS), must be chosen for considering the dataset; this process has already been explored in the literature (Park & Choi, 2009; Turk & Pentland, 1991). Then, the raw data implements a chosen method, which provides the selected significant features as the output. To solve the classification problem, these feature selection methods are usually used in the preprocessing procedure, and the selected features are applied to the classifier as the input. However, the proposed model employs a training set containing raw data as inputs and selected features as outputs to be implemented in the neural network. Through the use of this training set, the neural network learns using an ELM as a regression mode, as shown in Fig. 3. While operating the ELM, the only parameter is the number of nodes in the single hidden layer. Due to its fast learning speed and robustness with network architecture, it only considers bounded candidates of a number of hidden nodes between 1 and 100, making it able to determine the nodes in a few iterations (Huang et al., 2011). As this process determines the best number of hidden nodes, the neural network architecture is automatically determined and the functions are transferred following the ELM's function: the input to the hidden layer employs a log sigmoid transfer function, and the hidden layer to the output uses a linear transfer function, as shown in Fig. 3. Due to their connecting weights, the learned neural network estimates the characteristics of the chosen feature selection algorithms, which have properties similar to the feature selector.

2.2.2. Create second neural network that mimics the functions of the classifier using an ELM

The second step is the procedure to reflect the traits of the classification algorithms, such as an SVM and a C4.5, using selected features and class information. The learning process in this step is similar to the previous step. We choose one classification algo-

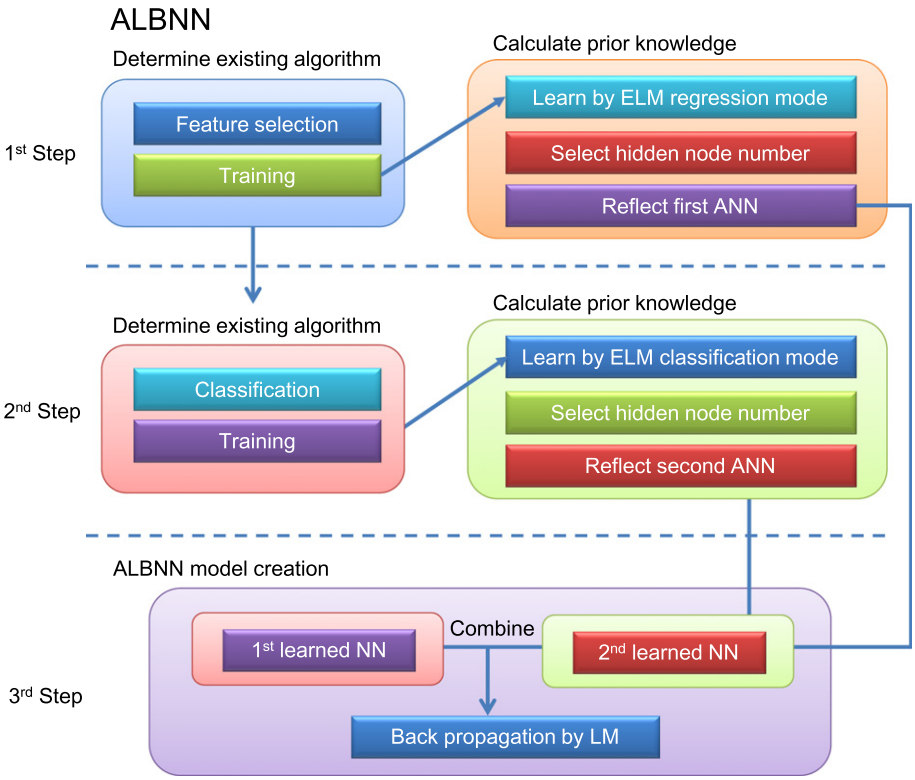

Fig. 2. ALBNN procedures.

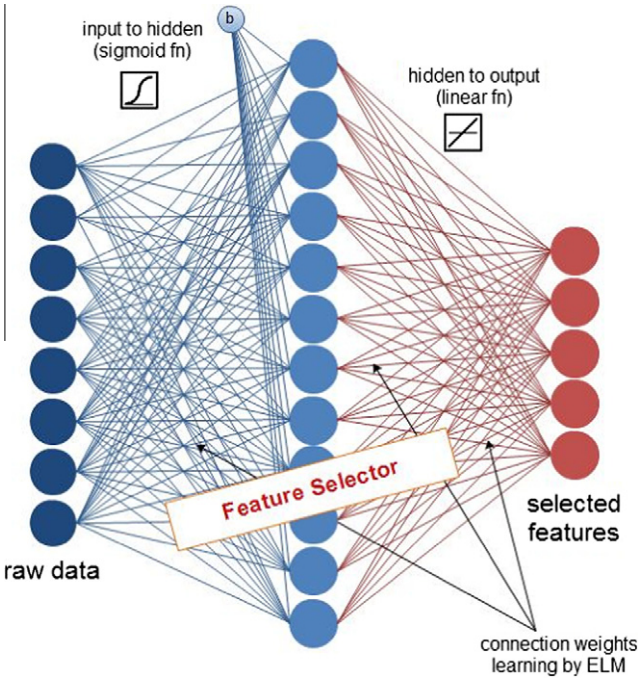

Fig. 3. Learned neural network as the feature selector.

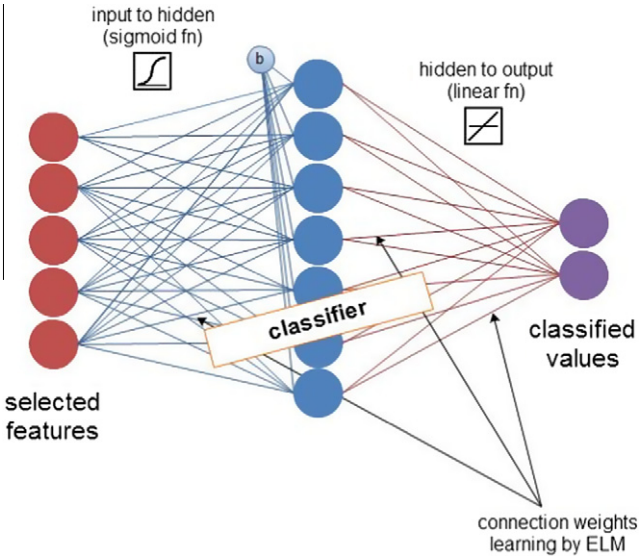

Fig. 4. Learned neural network as the classifier.

rithm to separate the selected features, which were determined by the feature selection algorithm in the first step, by a hyperplane as they process to fit the parameter for the considered dataset. As shown in Fig. 4, in order for the second neural network to learn, we employ the selected feature as the input and the classification information as the output in a training set. This second neural network also learns using an ELM in the classification mode because the class information contains categorical values. This network becomes the classifier, which mimics the properties of the classification algorithm.

2.2.3. Incorporate the first and second neural networks and learn using an LM

The third step is to construct the final model by integrating the first and second independently learned neural networks to improve the classification accuracy. Identical selected features are used as the outputs in the first and as the inputs in the second neu-

ral network so that the number of nodes is equal between the output nodes of the first neural network and the input nodes of the second neural network. Therefore, they can be connected sequentially as shown in Fig. 5. This procedure creates one neural network with links that already contain properties from the previous feature selection and classification within the neural network connection weights. Finally, the neural networks learn to achieve the main objective of improving the performance of classification. We consider the raw data as the input and the target class as the output in a training set for the last learning step. The combined neural networks do not learn using an ELM. The ELM algorithm not only operates on the single hidden-layer neural network, but also over-fits the learned neural network when the LM algorithm modifies them again. Through the ELM in the first and second steps, the combined neural networks have already determined the architecture and estimated the prior knowledge; thus, we implement an LM algorithm in the last learning step. The LM performs well to fit the target function through the iterative optimization procedure without requiring a significant amount of time to complete the process under the fixed architecture.

In this study, the ALBNN employs a novel technique by incorporating prior knowledge estimated from characteristics of other learning methods, and the technique maintains advantages consistent with the feature selection method. This process produces a more correctly approximated target hypothesis by considering the interaction between the two kinds of neural networks and by minimizing the individual errors. Therefore, the process can help to improve the generalization accuracy in classification problems and produce new relevant features.

2.3. Example study

A simulation study is performed to illustrate the effectiveness of an ALBNN using R software, which is the most widely used statistical software. We generate two classes of data that follow different multivariate normal distributions and created overlapping areas with 1000 data points each by modifying the parameters of distribution, as follows.

$$X_{\text{class1}} \sim \mathcal{N}(\mu_1, \Sigma_1)$$

$$X_{\text{class2}} \sim \mathcal{N}(\mu_2, \Sigma_2)$$

where

$$\mu_1 = \begin{pmatrix} 1 \\ 0 \\ \frac{3}{2} \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} \frac{5}{2} & \frac{2}{5} & \frac{1}{10} \\ \frac{2}{5} & \frac{7}{10} & \frac{3}{10} \\ \frac{1}{10} & \frac{3}{10} & \frac{3}{10} \end{pmatrix}.$$

$$\mu_2 = \begin{pmatrix} 2 \\ \frac{1}{10} \\ -\frac{3}{10} \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} \frac{7}{10} & \frac{2}{5} & \frac{1}{5} \\ \frac{2}{5} & \frac{3}{2} & \frac{1}{2} \\ \frac{1}{5} & \frac{1}{2} & \frac{17}{10} \end{pmatrix}$$

Fig. 6 shows the data plots of the different classes in three dimensions. Separating the data by a hyperplane is difficult. If the classification method implements those datasets directly, the data are vulnerable to misclassification in the mixed areas. When we classify this data using an SVM, the accuracy of the classification is only 87.21%. These kinds of data require extraction of relevant features from the original data.

Fig. 7 shows the features extracted from the PCA. The PCA produces new features to be reflected and maximizes variances between attributes but ignores class information. Even after applying the PCA to this data, significant overlapping still exists. When the SVM conducts a classification of two selected features from the PCA, it provides an 88.43% classification accuracy.

The ALBNN produces new modified features that are more appropriate for multivariate classification because it considers not only the target class value but also class information, the interaction between feature selection and classification, and the correlation between variables. This additional information is reflected in these new features through the three learning steps. Figs. 8–10 show the changes in features during the learning of a combined neural network. Fig. 8 shows the features obtained from the learning of the ALBNN in the 13th epoch. Fig. 9 shows the features obtained from the learning of the ALBNN in the 34th epoch. Fig. 10

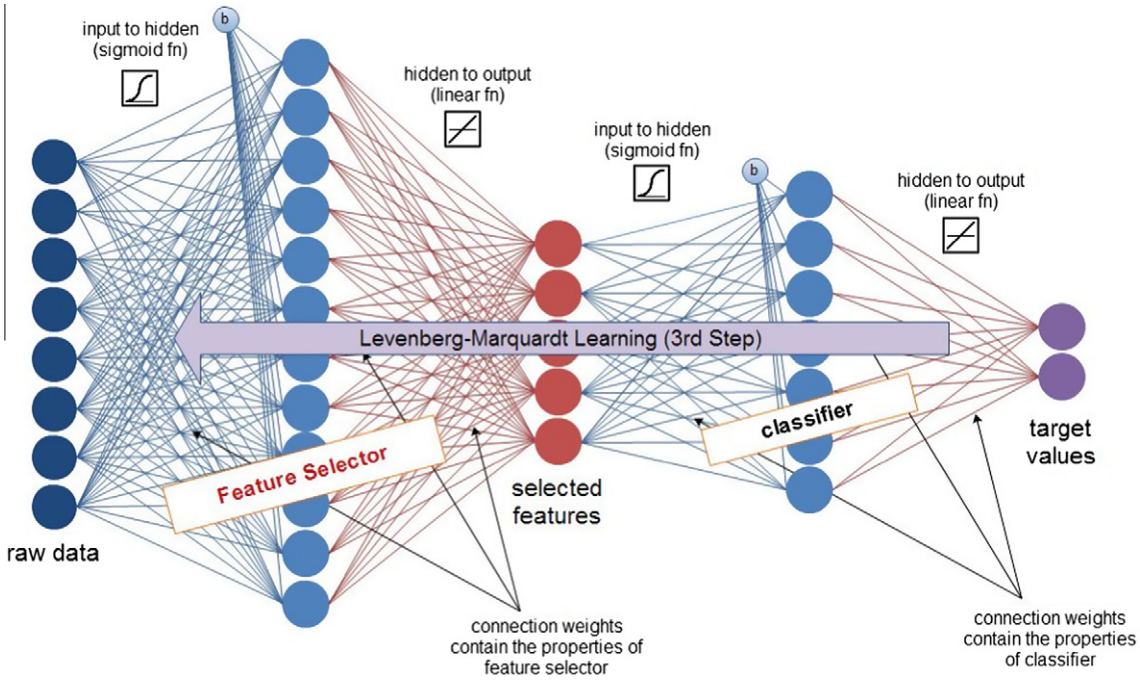

Fig. 5. Two incorporated neural networks.

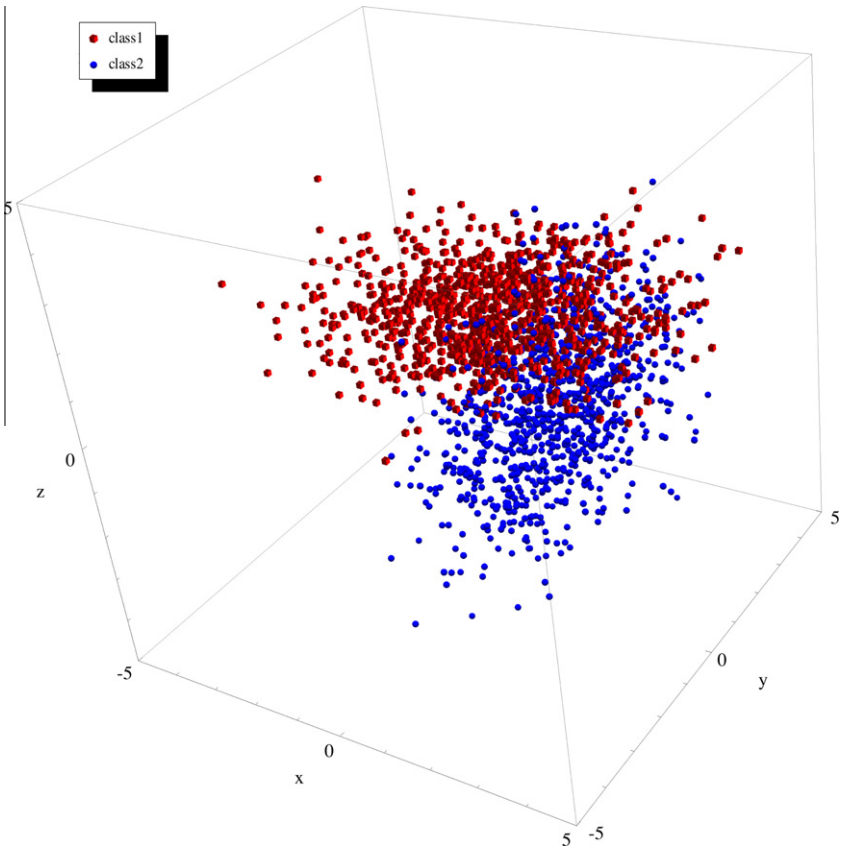

Fig. 6. Two-class simulated data displayed in 3D.

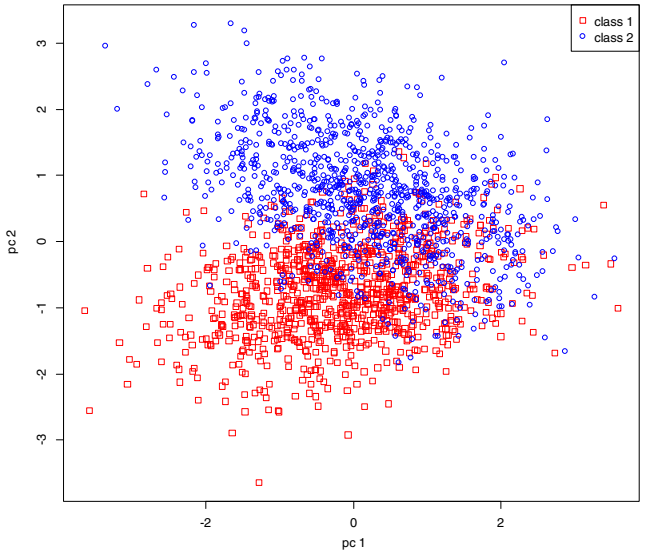

Fig. 7. Two features (PCs) from PCA.

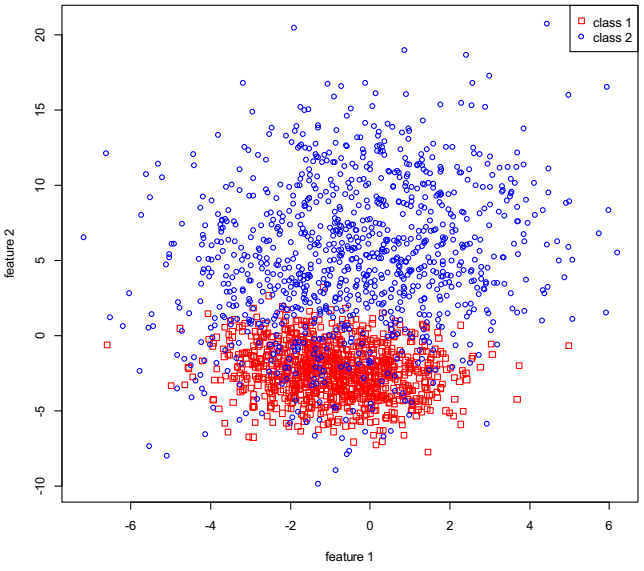

Fig. 8. New features from ALBNN in 13th epoch.

shows the features obtained from the learning of the ALBNN in the 49th epoch.

We observe the reduction of overlapping areas as the neural network learns. Compared to principal components (PCs), the new features produced from the 49th learning epoch of the ALBNN are more suitable for the classification problem. The classification accuracy was 92.89% in the final learned model until the 49th epoch. This indicates an improvement in the classification performance by the ALBNN compared to previous algorithms.

3. Experiments

In this section, the performance of the ALBNN learning algorithm is compared with traditional algorithms, such as the ANN, support vector machine (SVM), and C4.5. The experiments by an ELM and the back propagation by an LM algorithm were carried out in MATLAB 2010b and the feature selections and classifications are conducted in the R environment running in a Core 2 Duo, 3.00-GHz CPU with a 4-GB RAM because the R

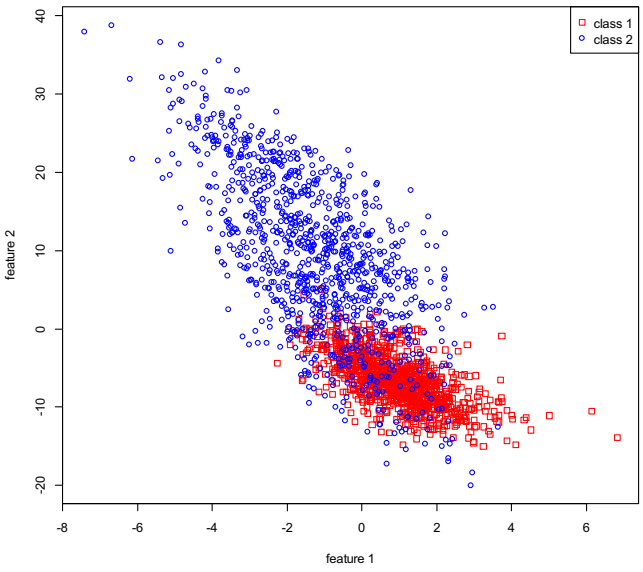

Fig. 9. New features from ALBNN in 34th epoch.

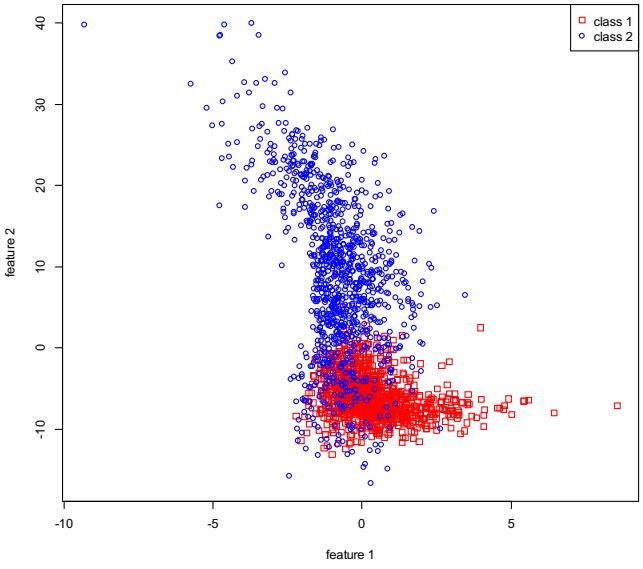

Fig. 10. New features from ALBNN in 49th epoch.

software does not provide neural network learning by an LM algorithm.

3.1. Design of experiments

We experimented with several data sets for real world classification at the University of California, Irvine (UCI) repository. Table 1 displays the detailed information of the datasets. Among the various datasets, we studied medical datasets and process datasets, which have many attributes that affect the results and follow different distributions. Identifying the high accuracy in those data sets is crucial, specifically the datasets have different ratios between classes in order to reflect the properties of abnormal conditions. Moreover, we also explore multi-class datasets such as heart disease, glass, and steel plates faults in order to verify the effects of an ALBNN.

Cross validation was used to evaluate the performance of algorithms in the classification problem because it eliminates results

Table 1
Description of datasets.

Datasets	Instances	Attributes	Classes
Parkinsons	195	22	2
Spectf	267	44	2
Horse colic	368	20	2
Pima indians diabetes	768	8	2
WDBC	569	30	2
Ionosphere	351	33	2
Hepatitis	137	16	2
Sonar	208	60	2
Heart disease	294	13	5
Glass	214	9	6
Steel plates fault	1941	27	7

that depend on the dataset. Hence, we conducted a 5-fold cross validation (CV) in this experiment. The m-fold CV operates in the following manner. Datasets are arbitrarily divided into m partitions without overlapping. The m-fold CV reserve one group among the m partitions as the testing dataset and $m-1$ groups utilizes for the training datasets. This procedure is repeated for all m partitions to evaluate the pertinent performance. The classification accuracy is the performance measure of this experiment.

In the first and second steps of the ALBNN, the node numbers of the hidden layer in each neural network was required to be determined as the ELM learns. The best node numbers in a hidden layer were selected between 1 and 100 through the mean squared errors (MSEs) in the first neural network by the ELM in regression mode and classification accuracy in the second neural network by the ELM in classification mode. After following all connection weights, the last learning step of the ALBNN was conducted by the LM algorithm.

3.2. Comparison algorithms

To evaluate the performance of the ALBNN, we compared it to a PCA and LARS as the feature selection technique and a SVM and C4.5 as the classification algorithm. Although a variety of learning methods are presented in the literature (Burges, 1998; Efron, Hastie, Johnstone, & Tibshirani, 2004; Paliwal & Kumar, 2009; Quinlan, 1993), we compared popular techniques because they have been widely used in real applications.

For the feature selection techniques, we first employed the PCA analysis. This feature extracting technique transforms the axis to produce the maximum amount of variability in the data as possible (Turk & Pentland, 1991). A few principal components selected as features have the best capacity to explain the variance of data and to retain the characteristics of the data while minimizing information loss of the data. This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced. Therefore, it has been widely used as a feature selection technique in high-dimensional data.

Another important model selection approach is to determine the best subset of features from a set of feature vectors for high-dimensional data. More specifically, the approach obtains a reduced model space most suitable for describing the data. The LARS algorithm, developed by Efron et al. (2004), has recently been used as a feature selection technique due to its computational efficiency in a full piece-wise step. It is similar to forward stepwise regression, but instead of including variables, the estimated parameters are increased in an angular direction equal to each parameter's correlation with the residual; thus, they become equally correlated (Efron et al., 2004). By utilizing those estimated parameters, a LARS provides significance levels between zero and one describing each variable's impact on the target values. In this experiment, we considered the significant feature using a threshold of 0.3 for classifi-

cation. In the Monte-Carlo trials, this determining threshold has shown to be quite reliable for feature selection.

In the classification techniques, the SVM was described as early as 1995 by Cortes and Vapnik and has been widely applied to the two class classification problems. The SVM converts linearly non-separable data to high-dimensional data for separation and classifies the data by quadratic programming. It is a statistical learning method to calculate the hyperplane for maximizing margins and has a good capacity for classification (Burges, 1998). Tree-based models also are widely used for classification and practical methods for intelligent decision making. The two general phases are tree growing and pruning. In the growing phase, splitting points are sought for all possible hypothesis spaces and then the trees are cut back to identify the pertinent size of the trees to avoid over-fitting in the pruning phase. The most popular algorithm in literature for building decision trees is C4.5 (Quinlan, 1993), an extension of the ID3 algorithm using the concept of information entropy. These post-pruning methods have been refined to determine the correct size of trees and provide minimum errors (Lim, Loh, & Shih, 2000).

Moreover, if the dataset has a low dimension, a feature selection procedure may not be required as the pre-processing step to extract the relevant features. Therefore, an ALBNN compares the results of the classification algorithm when it learns the single hidden-layer neural network using an LM without the feature selection procedure as well.

3.3. Experimental results

Table 2 shows the results of the classification using the algorithms described in Section 3.2. The results listed on the following tables are the mean and standard deviation (SD) of the testing accuracy. We first compared performances of popular classification algorithms, which are the C4.5, SVM, ELM, ANN (LM), and ALBNN proposed by Yoon and Baek (2011). The ANN (LM) completed the general single hidden-layer neural network learning using the LM algorithm, and Yoon and Baek (2011) employed the LM for three steps with the combinations of PCA and SVM. As shown in Table 2, the ELM, SVM, and Yoon and Baek (2011) perform well. The general ANN using the LM algorithm underperformed or contained a high variance of accuracy. Moreover, Yoon and Baek (2011) did not provide the best performance overall datasets with the disadvantages of learning time.

Table 3 shows the training and test classification results of the original ELM and ELM (LM). The ELM (LM) completed the single hidden layer neural network learning using an ELM and was modified by the LM algorithm again. Although training accuracies were significantly improved, test performances decreased, indicating that these learning strategies had over-fitting issues. Therefore, the ELM could not simply be improved through additional iterative tuning procedures.

Tables 4 and 5 show the comparisons of the ALBNN with other algorithms. We first compared the SVM and the SVM classifier after extracting features from the PCA or LARS, which we identified as (PCA + SVM), in which the SVM classifier is applied after the PCA, and (LARS + SVM), in which the SVM classifier is applied after the LARS. Feature selection techniques were successfully performed in these datasets because (PCA + SVM) and (LARS + SVM) exhibited better classification than the SVM alone. In this paper, we identified ALBNN (A + B) and an ALBNN that employs the A algorithm as the feature selection algorithm and the B method as the classifier. For example, ALBNN (PCA + SVM) is an ALBNN model that implements a PCA as the first learning step and an SVM as the second learning step. The proposed method outperformed the benchmarks. Specifically, the ALBNN (PCA + SVM) performed well in

Table 2
Testing accuracy of C4.5, SVM, ELM, ELM (LM), ANN (LM) and Yoon and Baek (2011).

Datasets	C4.5		SVM		ELM		ANN (LM)		Yoon and Baek (2011) (PCA + SVM)	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Parkinsons	0.831	0.053	0.882	0.039	**0.918***	0.056	0.838	0.075	0.881	0.012
Spectf	0.745	0.013	0.782	0.056	**0.820***	0.087	0.718	0.038	0.813	0.023
Horse colic	0.663	0.032	0.676	0.068	**0.720***	0.044	0.667	0.036	0.691	0.024
PID[a]	0.727	0.044	0.760	0.033	0.781	0.016	0.721	0.014	**0.786***	0.012
WDBC	0.917	0.031	0.975	0.010	0.975	0.007	**0.979***	0.009	0.958	0.006
Ionosphere	0.903	0.031	0.934	0.037	0.897	0.024	0.910	0.029	**0.962***	0.008
Hepatitis	0.789	0.045	0.846	0.055	**0.890***	0.063	0.821	0.048	0.847	0.028
Sonar	0.764	0.046	**0.831***	0.069	0.784	0.036	0.709	0.092	0.712	0.132
Heart disease	0.660	0.025	0.660	0.040	**0.680***	0.074	0.631	0.122	0.617	0.215
Glass	**0.710***	0.035	0.650	0.090	0.692	0.051	0.668	0.089	0.661	0.098
SPF[b]	0.752	0.025	**0.764***	0.019	0.728	0.013	0.714	0.077	0.701	0.091

[a] Pima indians diabetes.
[b] Steel plates fault.

Table 3
Training and testing accuracy of ELM and ELM (LM).

Datasets	ELM train		ELM (LM) train		ELM test		ELM (LM) test	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Parkinsons	0.983	0.006	**1.000***	0.000	**0.918***	0.056	0.851	0.038
Spectf	0.841	0.022	**0.901***	0.029	**0.820***	0.087	0.713	0.069
Horse colic	0.768	0.013	**0.831***	0.060	**0.720***	0.044	0.554	0.035
Pima indians diabetes	0.790	0.005	**0.830***	0.027	**0.781***	0.016	0.592	0.031
WDBC	0.978	0.005	**0.992***	0.007	**0.975***	0.007	0.611	0.054
Ionosphere	**0.949***	0.008	0.934	0.148	**0.897***	0.024	0.503	0.121
Hepatitis	**0.933***	0.020	0.900	0.114	**0.890***	0.063	0.741	0.079
Sonar	0.978	0.005	**1.000***	0.000	**0.784***	0.036	0.776	0.052
Heart disease	0.809	0.008	**0.881***	0.074	**0.680***	0.074	0.502	0.090
Glass	**0.841***	0.018	0.789	0.049	**0.692***	0.051	0.661	0.027
Steel plates fault	0.765	0.007	**0.944***	0.024	**0.728***	0.013	0.723	0.026

Table 4
Testing accuracy of SVM, PCA + SVM, LARS + SVM, Yoon and Baek (2011), and ALBNN combinations.

Datasets	SVM	PCA + SVM	LARS + SVM	Yoon and Baek (2011) (PCA + SVM)	Yoon and Baek (2011) (LARS + SVM)	ALBNN (PCA + SVM)	ALBNN (LARS + SVM)
Parkinsons	0.882[c] (0.039)[d]	0.887 (0.072)	0.882 (0.050)	0.881 (0.012)	0.903 (0.066)	0.954 (0.022)	**0.964*** (0.053)
Spectf	0.782 (0.056)	0.808 (0.077)	0.771 (0.045)	0.813 (0.023)	0.811 (0.056)	**0.898*** (0.037)	0.894 (0.043)
Horse colic	0.676 (0.068)	0.698 (0.054)	0.633 (0.074)	0.691 (0.024)	0.768 (0.074)	**0.784*** (0.080)	0.784 (0.094)
PID[a]	0.760 (0.033)	0.766 (0.041)	0.770 (0.045)	0.786 (0.012)	**0.795*** (0.044)	0.778 (0.036)	0.768 (0.065)
WDBC	0.975 (0.010)	0.958 (0.020)	0.958 (0.016)	0.958 (0.006)	0.961 (0.024)	**0.988*** (0.010)	0.986 (0.005)
Ionosphere	0.934 (0.037)	0.952 (0.019)	0.937 (0.039)	0.962 (0.008)	0.914 (0.035)	**0.971*** (0.027)	0.954 (0.028)
Hepatitis	0.846 (0.055)	0.876 (0.057)	0.840 (0.055)	0.847 (0.028)	0.882 (0.048)	**0.926*** (0.069)	0.896 (0.041)
Sonar	0.831 (0.069)	0.827 (0.037)	0.826 (0.056)	0.712 (0.132)	0.725 (0.067)	**0.919*** (0.032)	0.900 (0.064)
Heart disease	0.660 (0.040)	0.677 (0.032)	0.639 (0.045)	0.617 (0.215)	0.719 (0.041)	**0.729*** (0.043)	0.712 (0.080)
Glass	0.650 (0.090)	0.636 (0.094)	0.668 (0.085)	0.661 (0.098)	**0.773*** (0.076)	0.767 (0.074)	0.716 (0.058)
SPF[b]	0.764 (0.019)	0.730 (0.030)	0.646 (0.028)	0.701 (0.091)	0.819 (0.042)	**0.823*** (0.021)	0.813 (0.017)

[a] Pima indians diabetes.
[b] Steel plates fault.
[c] Mean.
[d] Standard deviation.

Table 5
Testing accuracy of C4.5, PCA + C4.5, LARS + C4.5, Yoon and Baek (2011), and ALBNN combinations.

Datasets	C4.5	PCA + C4.5	LARS + C4.5	Yoon and Baek (2011) (PCA + C4.5)	Yoon and Baek (2011) (LARS + C4.5)	ALBNN (PCA + C4.5)	ALBNN (LARS + C4.5)
Parkinsons	0.831[c] (0.053)[d]	0.846 (0.054)	0.836 (0.082)	0.913 (0.050)	0.939 (0.039)	**0.954*** (0.049)	0.949 (0.018)
Spectf	0.769 (0.067)	0.791 (0.048)	0.783 (0.013)	0.845 (0.059)	0.876 (0.034)	**0.898*** (0.029)	0.894 (0.051)
Horse colic	0.677 (0.038)	0.668 (0.033)	0.660 (0.069)	0.733 (0.057)	0.797 (0.033)	0.751 (0.079)	**0.824*** (0.066)
PID[a]	0.727 (0.044)	0.729 (0.049)	0.734 (0.030)	0.727 (0.077)	0.765 (0.017)	**0.770*** (0.053)	0.751 (0.076)
WDBC	0.917 (0.031)	0.949 (0.011)	0.942 (0.010)	0.958 (0.014)	0.944 (0.031)	**0.991*** (0.009)	0.988 (0.005)
Ionosphere	0.903 (0.031)	0.883 (0.044)	0.895 (0.022)	0.929 (0.042)	0.926 (0.024)	0.923 (0.039)	**0.934*** (0.048)
Hepatitis	0.789 (0.045)	0.824 (0.043)	0.789 (0.058)	0.896 (0.080)	**0.896*** (0.048)	0.889 (0.094)	0.889 (0.074)
Sonar	0.764 (0.046)	0.784 (0.027)	0.760 (0.065)	0.763 (0.086)	0.715 (0.025)	0.829 (0.142)	**0.929*** (0.075)
Heart disease	0.660 (0.025)	0.609 (0.023)	0.626 (0.038)	**0.758*** (0.071)	0.730 (0.098)	0.715 (0.047)	0.739 (0.023)
Glass	0.710 (0.035)	0.710 (0.110)	0.682 (0.075)	**0.766*** (0.049)	0.714 (0.100)	0.698 (0.029)	0.684 (0.075)
SPF[b]	0.752 (0.025)	0.659 (0.027)	0.742 (0.035)	0.807 (0.032)	0.799 (0.022)	**0.814*** (0.025)	0.794 (0.009)

[a] Pima indians diabetes.
[b] Steel plates fault.
[c] Mean.
[d] Standard deviation.

most cases. Moreover, the processing of some datasets, such as spectf and horse colic, exhibited significant improvement with a performance above 10%. An improving effect on the classification performance occurred via the additional feature modification of the ALBNN, which operated to more correctly adjust for the target.

Furthermore, we also tested some multiclass datasets, such as heart disease, glass, and steel plates faults. The effectiveness of the ALBNN was also demonstrated in the multiclass areas, as shown in Table 4. In order to implement multiclass classification, we considered the output node number of the second neural network as the same as the class number. The ELM analytically determined the output weights from the hidden layer with a linear transfer function in order to maintain an equal number of output nodes and classes, which performed better in multiclass problems. The results show that all three kinds of datasets were significantly improved by the ALBNN, verifying that the proposed method also performed well in multiclass classification problems.

Similar to the previous experiment, we investigated the C4.5 algorithm in conjunction with the same combinations. In the same manner, the ALBNN improved the classification performance of

Table 6
First and second high-performing algorithms.

Datasets	First rank algorithm	Second rank algorithm
Parkinsons	ALBNN (LARS + SVM)	ALBNN (PCA + SVM)
Spectf	ALBNN (PCA + C4.5)	ALBNN (PCA + SVM)
Horse colic	ALBNN (LARS + C4.5)	Yoon and Baek (2011) (LARS + C4.5)
Pima indians diabetes	Yoon and Baek (2011) (LARS + SVM)	Yoon and Baek (2011) (PCA + SVM)
WDBC	ALBNN (PCA + C4.5)	ALBNN (LARS + C4.5)
Ionosphere	ALBNN (PCA + SVM)	Yoon and Baek (2011) (PCA + SVM)
Hepatitis	ALBNN (PCA + SVM)	ALBNN (LARS + SVM)
Sonar	ALBNN (LARS + C4.5)	ALBNN (PCA + SVM)
Heart disease	ALBNN (LARS + C4.5)	ALBNN (PCA + SVM)
Glass	Yoon and Baek (2011)(PCA + C4.5)	ALBNN (PCA + SVM)
Steel plates fault	ALBNN (PCA + SVM)	ALBNN (PCA + C4.5)

Table 7
Average training times (s) of each algorithm.

Datasets	C4.5	SVM	PCA + SVM	LARS + SVM	PCA + C4.5	LARS + C4.5	Yoon and Baek (2011) (PCA + SVM)	ALBNN (PCA + SVM)	ALBNN (LARS + SVM)
Parkinsons	0.108	0.022	0.010	0.016	0.096	0.106	3693.859	56.791	110.979
Spectf	0.170	0.052	0.028	0.032	0.066	0.112	2647.054	71.059	427.983
Horse colic	2.142	0.046	0.046	0.034	0.106	0.134	2580.466	29.787	51.200
PID[a]	0.398	0.142	0.130	0.110	0.168	0.324	1237.885	24.040	15.934
WDBC	0.260	0.066	0.046	0.044	0.178	0.108	5208.235	332.132	119.681
Ionosphere	0.148	0.054	0.032	0.050	0.986	0.134	12589.838	73.099	246.479
Hepatitis	0.088	0.014	0.008	0.016	0.064	0.074	448.737	5.919	6.293
Sonar	0.162	0.040	0.018	0.032	0.098	0.132	6615.126	91.750	165.586
HD[b]	0.090	0.044	0.022	0.040	0.094	0.086	886.412	31.384	56.020
Glass	0.110	0.030	0.028	0.022	0.128	0.078	1062.740	14.854	6.006
SPF[c]	0.742	1.014	0.836	0.900	0.288	0.372	16910.270	321.827	306.380

[a] Pima indians diabetes.
[b] Heart disease.
[c] Steel plates fault.

both two-class and multiclass datasets with the exception of only one dataset. Although the C4.5 algorithm provided the best performance in the glass dataset, the results of the other algorithms for this dataset, including the ALBNN, are similar with 1.2–2.6% differences. However, the results of the datasets excluding the glass dataset demonstrate that the ALBNN efficiently completed the improved classification accuracy. These results demonstrate the effectiveness of the proposed method and its robustness to the class types.

In order to implement the ALBNN in classification areas, verification is required whether the feature selection is useful or not for considering the dataset. When the selected feature loses any salient information, the ELM can possibly be misled and thus causes the ALBNN to perform poorly. In Table 6, the first and second high-performing algorithm is in each dataset. Excluding just two datasets, the proposed model exhibited the best accuracy. We observed that little difference occurred in terms of the best performing combination. The ALBNN using the PCA performed well overall because it did not significantly lose information in the feature selection step. The ALBNN also achieved significantly improved classification accuracy in the multiclass problems.

Compared with the process of Yoon and Baek (2011), in which the LM was employed for all three steps which took a long training time, the current process is faster in determining the node numbers. In our experiments, the learning phase of the ELM in the first and second neural networks was completed in seconds for many datasets and was not sensitive to the node numbers in the hidden layer. In this study, the ALBNN still used the LM algorithm in the third step; however, the learning time was reduced to less than 428 seconds. As shown in Table 7, the standard deviation is smaller than the benchmark of the ANN, indicating that the ALBNN is more stable and converged to the narrow hypothesis spaces by using the ELM algorithm first.

4. Conclusions

In our research, we propose a new method of neural networks, called the ALBNN, to integrate the procedures of feature selection and classification to improve the performance of classification. The essential idea is to incorporate estimated prior knowledge from traits of other learning algorithms, which is based on the theory of KBANNs. The prior knowledge could provide a better starting point to estimate the target function in a neural network, and the model learns again by an LM in order to fit the original targets.

In general, a neural network estimates complex nonlinear functions well but requires a procedure to determine the best architecture, which results in a significant loss of time. Many researchers have explored this concept as an optimization of work. However, the ALBNN employs an ELM to learn the neural network so that parameters can be determined faster and as they process, thereby eliminating time-consuming work. Moreover, the proposed method also produces new relevant features to fit classification problems. In the learning step of the LM, the method can modify the selected features that are all reflected in the classification information, the interaction between the feature selection and classification, and the correlation between the variables. We compared the ALBNN with several combinations of the PCA, LARS, SVM, and C4.5, which have been widely used for feature selection and classification. The experimental results demonstrated that the ALBNN improved the classification accuracy in most cases. The ALBNN using the PCA performed well in two class datasets and multiclass datasets.

An important advantage of the ALBNN is that it can be implemented in various algorithms as an extension. In order to apply an ALBNN to improve the classification accuracy, only the result sets that contain raw data, selected features, and classified information generated from any other learning algorithm are required.

Further study on the effects of various combinations of techniques remains. The investigated techniques could then be applied to high-dimensional datasets.

Acknowledgement

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the IT R&D Infrastructure Program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2011-(B1110-1101-0002)). This research was supported by National IT Industry Promotion Agency (NIPA) under the program of Software Engineering Technologies Development and Experts Education.

References

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2*, 121–167.

Cagdas, H. A. (2011). A new architecture selection method based on tabu search for artificial neural networks. *Expert Systems with Applications, 38*(4), 3287–3293.

Chauvin, Y., & Rumelhart, D. E. (1995). *Back Propagation: theory, architecture, and applications*. New Jersey: Lawrence Erlbaum Associates.

Cortes, C., & Vapnik, V. (1995). Support vector network. *Machine Learning, 20*, 273–297.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics, 32*(2), 407–499.

Egrioglu, E., Aladag, C. H., & Gunay, S. (2008). A new model selection strategy in artificial neural network. *Applied Mathematics and Computation, 195*(2), 591–597.

Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks, 5*(6), 989–993.

Huang, G.-B., Zhou, H., Ding, X., & Zhang, R. (2011). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on systems, man, and cybernetics, 99*, 1–17.

Huang, G.-B., Zhu, Q.-Y., & Siewa, C.-K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing, 70*, 489–501.

Kabir, M. M., Islam, M. M., & Murase, K. (2010). A new wrapper feature selection approach using neural network. *Neruocomputing, 73*, 3273–3283.

Kimes, D. S., Nelson, R. F., Manry, M. T., & Fung, A. K. (1998). Attributes of neural networks for extracting continuous vegetation variables from optical and radar measurements. *International Journal of Remote Sensing, 19*(14), 2639–2663.

Lim, T.-S., Loh, W.-Y., & Shih, Y.-S. (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning, 40*, 203–228.

Paliwal, M., & Kumar, U. A. (2009). Neural networks and statistical techniques: A review of applications. *Expert Systems with Applications, 36*, 2–17.

Park, M. S., & Choi, J. Y. (2009). Theoretical analysis on feature extraction capability of class-augmented PCA. *Journal of Pattern Recognition, 42*, 2353–2362.

Pudil, P., Novovicova, J., & Kittler, J. (1994). Floating search methods in feature selection. *Pattern Recognition Letters, 15*(11), 1119–1125.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. CA: Morgan Kaufmann.

Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics, 23*(19), 2507–2517.

Sahin, S., Tolun, M. R., & Hassanpour, R. (2012). Hybrid expert systems: A survey of current approaches and applications. *Expert Systems with Applications, 39*(4), 4609–4617.

Sarkar, I. N., Planet, P. J., Bael, T. E., Stanley, S. E., Siddall, M., & DeSall, R. (2002). Characteristic attributes in cancer microarrays. *Journal of Biomedical Informatics, 35*(2), 111–122.

Towell, G. G., & Shavlik, J. W. (1994). Knowledge based artificial neural networks. *Artificial Intelligence, 70*(1), 119–165.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognitions. *Journal of Cognitive Neuroscience, 3*, 71–86.

Yoon, H., & Baek, J.-G. (2011). Feature selecting and classifying integrated neural network algorithm for multi-variate classification. *IE Interfaces, 24*(2), 97–104.

Wang, T.-Y., & Huang, C.-Y. (2007). Applying optimized BPN to a chaotic time series problem. *Expert Systems with Applications, 32*(1), 193–200.